INFORMATION ABOUT PRINCIPAL INVESTIGATORS/PROJECT DIRECTORS(PI/PD) and co-PRINCIPAL INVESTIGATORS/co-PROJECT DIRECTORS

Submit only ONE copy of this form **for each PI/PD and co-PI/PD** identified on the proposal. The form(s) should be attached to the original proposal as specified in GPG Section II.B. Submission of this information is voluntary and is not a precondition of award. This information will not be disclosed to external peer reviewers. ***DO NOT INCLUDE THIS FORM WITH ANY OF THE OTHER COPIES OF YOUR PROPOSAL AS THIS MAY COMPROMISE THE CONFIDENTIALITY OF THE INFORMATION.***

PI/PD Name: Cem Kaner

Gender: ☐ Male ☐ Female

Ethnicity: (Choose one response) ☐ Hispanic or Latino ☐ Not Hispanic or Latino

Race: (Select one or more)
- ☐ American Indian or Alaska Native
- ☐ Asian
- ☐ Black or African American
- ☐ Native Hawaiian or Other Pacific Islander
- ☐ White

Disability Status: (Select one or more)
- ☐ Hearing Impairment
- ☐ Visual Impairment
- ☐ Mobility/Orthopedic Impairment
- ☐ Other
- ☐ None

Citizenship: (Choose one) ☒ U.S. Citizen ☐ Permanent Resident ☐ Other non-U.S. Citizen

Check here if you do not wish to provide any or all of the above information (excluding PI/PD name): ☒

REQUIRED: Check here if you are currently serving (or have previously served) as a PI, co-PI or PD on any federally funded project ☐

Ethnicity Definition:
Hispanic or Latino. A person of Mexican, Puerto Rican, Cuban, South or Central American, or other Spanish culture or origin, regardless of race.

Race Definitions:
American Indian or Alaska Native. A person having origins in any of the original peoples of North and South America (including Central America), and who maintains tribal affiliation or community attachment.
Asian. A person having origins in any of the original peoples of the Far East, Southeast Asia, or the Indian subcontinent including, for example, Cambodia, China, India, Japan, Korea, Malaysia, Pakistan, the Philippine Islands, Thailand, and Vietnam.
Black or African American. A person having origins in any of the black racial groups of Africa.
Native Hawaiian or Other Pacific Islander. A person having origins in any of the original peoples of Hawaii, Guam, Samoa, or other Pacific Islands.
White. A person having origins in any of the original peoples of Europe, the Middle East, or North Africa.

WHY THIS INFORMATION IS BEING REQUESTED:

The Federal Government has a continuing commitment to monitor the operation of its review and award processes to identify and address any inequities based on gender, race, ethnicity, or disability of its proposed PIs/PDs. To gather information needed for this important task, the proposer should submit a single copy of this form for each identified PI/PD with each proposal. Submission of the requested information is voluntary and will not affect the organization's eligibility for an award. However, information not submitted will seriously undermine the statistical validity, and therefore the usefulness, of information recieved from others. Any individual not wishing to submit some or all the information should check the box provided for this purpose. (The exceptions are the PI/PD name and the information about prior Federal support, the last question above.)

Collection of this information is authorized by the NSF Act of 1950, as amended, 42 U.S.C. 1861, et seq. Demographic data allows NSF to gauge whether our programs and other opportunities in science and technology are fairly reaching and benefiting everyone regardless of demographic category; to ensure that those in under-represented groups have the same knowledge of and access to programs and other research and educational oppurtunities; and to assess involvement of international investigators in work supported by NSF. The information may be disclosed to government contractors, experts, volunteers and researchers to complete assigned work; and to other government agencies in order to coordinate and assess programs. The information may be added to the Reviewer file and used to select potential candidates to serve as peer reviewers or advisory committee members. See Systems of Records, NSF-50, "Principal Investigator/Proposal File and Associated Records", 63 Federal Register 267 (January 5, 1998), and NSF-51, "Reviewer/Proposal File and Associated Records", 63 Federal Register 268 (January 5, 1998).

List of Suggested Reviewers or Reviewers Not To Include (optional)

SUGGESTED REVIEWERS:
Not Listed

REVIEWERS NOT TO INCLUDE:

COVER SHEET FOR PROPOSAL TO THE NATIONAL SCIENCE FOUNDATION

PROGRAM ANNOUNCEMENT/SOLICITATION NO./CLOSING DATE/if not in response to a program announcement/solicitation enter NSF 01-2

NSF 00-126 **01/24/01**

FOR NSF USE ONLY

NSF PROPOSAL NUMBER

FOR CONSIDERATION BY NSF ORGANIZATION UNIT(S) (Indicate the most specific unit known, i.e. program, division, etc.)

IIS - ITR SMALL GRANTSEIA - ITR SMALL GRANTS

DATE RECEIVED	NUMBER OF COPIES	DIVISION ASSIGNED	FUND CODE	DUNS# (Data Universal Numbering System)	FILE LOCATION
				053396669	

EMPLOYER IDENTIFICATION NUMBER (EIN) OR TAXPAYER IDENTIFICATION NUMBER (TIN)

596046500

SHOW PREVIOUS AWARD NO. IF THIS IS
☐ A RENEWAL
☐ AN ACCOMPLISHMENT-BASED RENEWAL

IS THIS PROPOSAL BEING SUBMITTED TO ANOTHER FEDERAL AGENCY? YES ☐ NO ☒ IF YES, LIST ACRONYMS(S)

NAME OF ORGANIZATION TO WHICH AWARD SHOULD BE MADE

Florida Institute of Technology

AWARDEE ORGANIZATION CODE (IF KNOWN)

0014696000

ADDRESS OF AWARDEE ORGANIZATION, INCLUDING 9 DIGIT ZIP CODE

Florida Institute of Technology
Melbourne, FL. 32901

NAME OF PERFORMING ORGANIZATION, IF DIFFERENT FROM ABOVE

PERFORMING ORGANIZATION CODE (IF KNOWN)

ADDRESS OF PERFORMING ORGANIZATION, IF DIFFERENT, INCLUDING 9 DIGIT ZIP CODE

IS AWARDEE ORGANIZATION (Check All That Apply)
(See GPG II.C For Definitions) ☐ FOR-PROFIT ORGANIZATION ☐ SMALL BUSINESS ☐ MINORITY BUSINESS ☐ WOMAN-OWNED BUSINESS

TITLE OF PROPOSED PROJECT **ITR/SY+PE: IMPROVING THE EDUCATION OF SOFTWARE TESTERS**

REQUESTED AMOUNT	PROPOSED DURATION (1-60 MONTHS)	REQUESTED STARTING DATE	SHOW RELATED PREPROPOSAL NO., IF APPLICABLE
$ **499,668**	**36** months	**09/01/01**	

CHECK APPROPRIATE BOX(ES) IF THIS PROPOSAL INCLUDES ANY OF THE ITEMS LISTED BELOW

☒ BEGINNING INVESTIGATOR (GPG I.A)
☐ DISCLOSURE OF LOBBYING ACTIVITIES (GPG II.C)
☐ PROPRIETARY & PRIVILEGED INFORMATION (GPG I.B, II.C.6)
☐ NATIONAL ENVIRONMENTAL POLICY ACT (GPG II.C.9)
☐ HISTORIC PLACES (GPG II.C.9)
☐ SMALL GRANT FOR EXPLOR. RESEARCH (SGER) (GPG II.C.11)
☐ VERTEBRATE ANIMALS (GPG II.C.11) IACUC App. Date _____
☒ HUMAN SUBJECTS (GPG II.C.11)
Exemption Subsection _____ or IRB App. Date **01/23/01**
☐ INTERNATIONAL COOPERATIVE ACTIVITIES: COUNTRY/COUNTRIES INVOLVED _____
☐ HIGH RESOLUTION GRAPHICS/OTHER GRAPHICS WHERE EXACT COLOR REPRESENTATION IS REQUIRED FOR PROPER INTERPRETATION (GPG I.E.1)

PI/PD DEPARTMENT
Computer Science

PI/PD POSTAL ADDRESS
150 West University Blvd

Melbourne, FL 329016975
United States

PI/PD FAX NUMBER
321-727-8084

NAMES (TYPED)	High Degree	Yr of Degree	Telephone Number	Electronic Mail Address
PI/PD NAME **Cem Kaner**	**Ph.D**	**1984**	**321-674-7137**	**kaner@kaner.com**
CO-PI/PD				
CO-PI/PD				
CO-PI/PD				
CO-PI/PD				

CERTIFICATION PAGE

Certification for Principal Investigators and Co-Principal Investigators:

I certify to the best of my knowledge that:

(1) the statements herein (excluding scientific hypotheses and scientific opinions) are true and complete, and
(2) the text and graphics herein as well as any accompanying publications or other documents, unless otherwise indicated, are the original work of the signatories or individuals working under their supervision. I agree to accept responsibility for the scientific conduct of the project and to provide the required progress reports if an award is made as a result of this proposal.

I understand that the willful provision of false information or concealing a material fact in this proposal or any other communication submitted to NSF is a criminal offense (U.S.Code, Title 18, Section 1001).

Name (Typed)	Signature	Social Security No.*	Date
PI/PD **Cem Kaner**			
Co-PI/PD			
Co-PI/PD			
Co-PI/PD			
Co-PI/PD			

SSN's are confidential and are not displayed *ON FASTLANE SUBMISSIONS*

Certification for Authorized Organizational Representative or Individual Applicant:

By signing and submitting this proposal, the individual applicant or the authorized official of the applicant institution is: (1) certifying that statements made herein are true and complete to the best of his/her knowledge; and (2) agreeing to accept the obligation to comply with NSF award terms and conditions if an award is made as a result of this application. Further, the applicant is hereby providing certifications regarding debarment and suspension, drug-free workplace, and lobbying activities (see below), as set forth in Grant Proposal Guide (GPG), NSF 01-2. Willful provision of false information in this application and its supporting documents or in reports required under an ensuring award is a criminal offense (U. S. Code, Title 18, Section 1001).

In addition, if the applicant institution employs more than fifty persons, the authorized official of the applicant institution is certifying that the institution has implemented a written and enforced conflict of interest policy that is consistent with the provisions of Grant Policy Manual Section 510; that to the best of his/her knowledge, all financial disclosures required by that conflict of interest policy have been made; and that all identified conflicts of interest will have been satisfactorily managed, reduced or eliminated prior to the institution's expenditure of any funds under the award, in accordance with the institution's conflict of interest policy. Conflict which cannot be satisfactorily managed, reduced or eliminated must be disclosed to NSF.

Debarment Certification

(If answer "yes", please provide explanation.)

Is the organization or its principals presently debarred, suspended, proposed for debarment, declared ineligible, or voluntarily excluded from covered transactions by any Federal department or agency? Yes ☐ No ☒

Certification Regarding Lobbying

This certification is required for an award of a Federal contract, grant, or cooperative agreement exceeding $100,000 and for an award of a Federal loan or a commitment providing for the United States to insure or guarantee a loan exceeding $150,000.

Certification for Contracts, Grants, Loans and Cooperative Agreements

The undersigned certifies, to the best of his or her knowledge and belief, that:

(1) No federal appropriated funds have been paid or will be paid, by or on behalf of the undersigned, to any person for influencing or attempting to influence an officer or employee of any agency, a Member of Congress, an officer or employee of Congress, or an employee of a Member of Congress in connection with the awarding of any federal contract, the making of any Federal grant, the making of any Federal loan, the entering into of any cooperative agreement, and the extension, continuation, renewal, amendment, or modification of any Federal contract, grant, loan, or cooperative agreement.

(2) If any funds other than Federal appropriated funds have been paid or will be paid to any person for influencing or attempting to influence an officer or employee of any agency, a Member of Congress, an officer or employee of Congress, or an employee of a Member of Congress in connection with this Federal contract, grant, loan, or cooperative agreement, the undersigned shall complete and submit Standard Form-LLL, ''Disclosure Form to Report Lobbying,'' in accordance with its instructions.

(3) The undersigned shall require that the language of this certification be included in the award documents for all subawards at all tiers including subcontracts, subgrants, and contracts under grants, loans, and cooperative agreements and that all subrecipients shall certify and disclose accordingly.

This certification is a material representation of fact upon which reliance was placed when this transaction was made or entered into. Submission of this certification is a prerequisite for making or entering into this transaction imposed by section 1352, title 31, U.S. Code. Any person who fails to file the required certification shall be subject to a civil penalty of not less than $10,000 and not more than $100,000 for each such failure.

AUTHORIZED ORGANIZATIONAL REPRESENTATIVE	SIGNATURE	DATE
NAME/TITLE (TYPED)		**01/23/01**

TELEPHONE NUMBER	ELECTRONIC MAIL ADDRESS	FAX NUMBER

*SUBMISSION OF SOCIAL SECURITY NUMBERS IS VOLUNTARY AND WILL NOT AFFECT THE ORGANIZATION'S ELIGIBILITY FOR AN AWARD. HOWEVER, THEY ARE AN INTEGRAL PART OF THE INFORMATION SYSTEM AND ASSIST IN PROCESSING THE PROPOSAL. SSN SOLICITED UNDER NSF ACT OF 1950, AS AMENDED.

A. PROJECT SUMMARY

Software testing is a time-consuming, expensive, difficult, but essential area of work in the development of reliable software. Testing is not well taught at the university level, partially because the area is virtually ignored in the ACM/IEEE Computer Science curriculum guide (1991, 2001). Computer Science graduates are rarely well equipped to work in software testing groups, and there is no alternative university degree program for software testers. The result is a shortage of skilled labor in a key area of Information Technology.

This project will lay a foundation for significant improvements in the quality of academic and commercial courses in software testing.

Specific deliverables will include:

- A collection of examples of software errors, well-described, with screen shots, and suitable for use in class. The collection will be posted on the web, with a permanently assigned name and URL for each example, making the examples easy to reference.
- Identification and descriptions of specific skills involved in software testing. These will be published at conferences and in articles that are available on the web.
- Keyed to the skills of interest, sample exercises for students (for use as course assignments or for self-paced study) and a smaller separate collection for teachers (for possible use in exams). The student exercises will be available in practice books or on the web. The teachers' materials will be available in a teacher's manual or on the web.
- Free software versions of classroom-level versions of useful testing tools. A "classroom version" is intended to teach the concept of that kind of tool, and to handle tasks that are as complex as the student might run into as an undergraduate. Such a tool will often be useful in commercial testing but it is not as full-featured as tools designed for professional use. These tools will be published on the web.
- Research (and reports on the results of the research) on the usefulness of Whittaker's software fault model as an organizing structure for presenting a wide range of key testing techniques in class.
- Extension of Kaner & Bach's characterizations of testing styles/strategies, bringing together research and practitioner literature into a structure useful for presenting a wide range of key testing strategies in class.
- Ongoing workshops (and an email discussion group) on the teaching of software testing.

This is new ground. The skills used in the field are not well identified and there is no collection of exercises or drills designed to help polish those skills. Books on software testing rarely include exercises, and even those that do include some, don't include many. Nor is there a solid collection of well-described examples that are suitable for reuse in a classroom.

The Principal Investigator for this work is qualified in several ways. He is senior author of *Testing Computer Software*, which is the best selling book on software testing in the history of the field (the publisher John Wiley and Sons makes this assertion based on lifetime total sales of the book). He has taught courses on software testing in commercial environments several dozens of times and his course notes are licensed out to other commercial software testing instructors. He is Professor of Computer Sciences at the Florida Institute of Technology, which has a core strength in software testing. And he holds a doctorate in Experimental Psychology with a specialization in human perception and performance (including study of how students learn and how experts make decisions).

TABLE OF CONTENTS

For font size and page formatting specifications, see GPG section II.C.

Section		Total No. of Pages in Section	Page No.* (Optional)*
Cover Sheet (NSF Form 1207) (Submit Page 2 with original proposal only)			
A	Project Summary (not to exceed 1 page)	1	
B	Table of Contents (NSF Form 1359)	1	
C	Project Description (plus Results from Prior NSF Support) (not to exceed 15 pages) **(Exceed only if allowed by a specific program announcement/solicitation or if approved in advance by the appropriate NSF Assistant Director or designee)**	15	
D	References Cited	2	
E	Biographical Sketches (Not to exceed 2 pages each)	2	
F	Budget (NSF Form 1030, plus up to 3 pages of budget justification)	6	
G	Current and Pending Support (NSF Form 1239)	1	
H	Facilities, Equipment and Other Resources (NSF Form 1363)	1	
I	Special Information/Supplementary Documentation	0	
J	Appendix (List below.) **(Include only if allowed by a specific program announcement/ solicitation or if approved in advance by the appropriate NSF Assistant Director or designee)**		
	Appendix Items:		

*Proposers may select any numbering mechanism for the proposal. The entire proposal however, must be paginated. Complete both columns only if the proposal is numbered consecutively.

C. PROJECT DESCRIPTION

C.1 BACKGROUND

> ***"Inside Microsoft, Windows NT is typical in that we have more testers than we have developers. And of course, testers spend all their time doing testing and developers spend a substantial part of their time involved with the testing. So when you think of Microsoft people think of a software development organization, which we certainly are. But actually testing these systems and making sure that the variety of hardware and networking protocols all comes together is a huge expense, and it's very important that we do that before we put a system out into deployment."*** --Gates (1997)

The quality of common commercial software products is rarely good enough (Kaner & Pels, 1998). We hear routinely about serious failures in basic operations, error handling, usability, safety, and resistance to malicious attacks. Software testing provides some of the methods that development teams can use to discover weaknesses and improve the quality of their products. Software development and publishing companies often spend enormous amounts of money on software testing. I believe that they are achieving too small a return on their investment, partially because we (the academic community) are doing a poor job of providing relevant education to the people who will do the software testing.

Software companies are struggling to apply good software development methods to software testing. For example:

- Even though this is often ineffective, software test automation is primarily done at the user interface level. (For example, the most recent two books on automated software testing, Fewster & Graham, 1999, and Dustin et al., 1999, focus exclusively on user interface level testing.)
- Many test automation efforts fail because the test tool users don't follow basic good programming practices (such as designing their test code to be maintainable in the face of changes in the software under test). (*See, e.g.,* Kaner, 1997; Pettichord 2000 and the papers at his website, www.pettichord.com).
- Few testers know how to build state models, design tests using cause-effect graphs, use code coverage tools to check the extent of testing completed, use systematic methods for efficient testing of combinations of variables.[1]

A substantial part of the problem is that few graduates from current computer science programs know enough about software testing, test-related theory, or relevant tools to apply basic techniques and use basic testing tools without extensive training.[2]

---

1 For example, see Whittaker's (1997) and Robinson's (2000) discussions of stochastic, model based testing, which make it clear that this approach is useful but advanced. I'm not aware of publications that describe what testers in the field actually know. This is an assertion based on my experience managing test groups, consulting to, and teaching classes to, test groups at a wide range of software or computer hardware companies, and discussing staff knowledge and training with several other test managers. Since 1994, I've taught classes on software testing over 75 times. My course notes on testing, test automation, and the recruiting of software testers (a 1-day seminar that I've presented repeatedly at the Software Testing Analysis & Review Conferences) are available on request. See Kaner (2000a, 2000b) and Kaner & Hoffman (2000).

2 As to what Computer Science graduates learn in school, read the ACM/IEEE (1991) *Computing Curricula.* The amount of required study of testing techniques is trivial—a few hours over four years. The curriculum guide suggests one optional course, *Advanced Software Engineering,* that includes testing topics among many others,

There is enormous demand in industry for skilled software testers. Many talented individuals spend their career as software testers and test managers. At Microsoft, and at many other software publishers, the typical software development project includes as many software testers as programmers.[3] Despite this significant role of testers in the field:

- There is no bachelor's-level degree program in the United States for software testing. Only a few universities grant Masters or Doctoral degrees in Computer Science or Software Engineering for software-testing related work.
- There is no software testing professional society.
- There is no ACM or IEEE journal focused on software testing.
- There is no widely accepted certification program in software testing. A few organizations "certify" software testers, but their tests (and so many of the commercially available testing courses) focus on definitions of terms and descriptions of processes and practices. There is relatively little emphasis on the development of specific skills that people will use regularly on the job. For example, consider the American Society for Quality's Certified Software Quality Engineer (www.asq.org/standcert/certification/csqe1.html). A typical 27-hour review course for the software quality exam includes about 1-2 hours on software testing. The rest of the time is spent on standards, process models, vocabulary associated with software quality issues, metrics, etc. The exam itself covers relatively few testing skills and in little depth. On a 160 question 4-hour exam, about 10 questions will be on software testing, and these are fairly general. For example, in the 18-question Study Guide posted at www.asq.org/standcert/certification/csqe1.html#csqestudy, only two questions involve testing skills. Here they are:

  "When an audit team concludes that a finding demonstrates a breakdown of the quality management system, the finding should be documented as:

  a. a minor nonconformance

  b. a major nonconformance

  c. a deficiency

  d. an observation"

  and

  "A module includes a control flow loop that can be executed 0 or more times. The test which is most likely to reveal loop initialization defects executes the loop body

  a. 0 times

  b. 1 time

  c. 2 times

  d. 3 times."

  Successful completion of a test at this level is not persuasive evidence of competence in testing.

---

but the guide doesn't even mention the idea of a course focused on software testing. ACM/IEEE (2001) is not significantly improved in this respect.

3 Regarding Microsoft, see Cusumano & Selby (1995), Gates (1997). Last October, I facilitated a two-day meeting of senior test managers, called the Software Test Manager's Roundtable, which is held in conjunction with the Software Testing Analysis & Review Conference (STAR West) in San Jose, CA. The focus of the meeting was the allocation of work and budget between programming and testing teams. Some projects described at this meeting had many more testers than programmers.

C.2 OBJECTIVES

My longer-term goal is the development of an undergraduate curriculum in software testing, as a specialization within a computer science degree.

My primary objective for the work proposed here is the creation and dissemination of materials that will support the teaching of high-caliber, challenging undergraduate courses in software testing. As an additional benefit, many of these materials will prove useful to graduate students and others who are doing research in the field.

My specific objectives for the work covered in this application include:

- Identify skills involved in software testing.
- Identify types of exercises that support the development of specific testing skills.
- Create and publish a collection of reusable materials for exercises and tests.
- Create and publish a collection of examples of software errors.
- Build and publish open source versions of several software testing tools.
- Strengthen Whittaker's fault model as a vehicle for teaching testing techniques.
- Strengthen Kaner & Bach's overview of software testing paradigms as a vehicle for teaching testing strategy and helping students build a context for a wide array of techniques.
- Create a series of workshops that focus on the teaching of software testing.
- Prototype a web-based course on software testing.

Each of these objectives is discussed in detail below.

C.3 SKILLS INVOLVED IN SOFTWARE TESTING

Textbooks in mathematics, physics, and other sciences often include hundreds of problems for students to solve on their own as exercises. As students learn how to solve collections of problems, they come to better understand the thinking behind the problems. Many science students use additional exercise books, like the *Schaum's Outlines*, to extend their skills. These drill books are available for elementary and more advanced-level courses, such as *Schaum's Outline of Partial Differential Equations* (Duchateau & Zachmann, 1994) and Sveshnikov's *Problems in Probability Theory, Mathematical Statistics, and Theory of Random Functions* (1979) .

Some testing skills can be mastered by use of self-paced, independent drill. Here are five examples of drills that support development of testing skills:

- ***When the program fails, the tester must write a failure report that clearly and accurately describes the steps that led to the failure and the final observable result.*** Consider the following exercise: Tester #1 looks at a computer screen and writes a description of everything that is displayed on the screen. The description is passed to Tester #2, who has not seen the screen. Tester #2 draws a picture of the screen, based on Tester #1's description, and then passes the drawing back to Tester #1. Keep practicing this until Tester #1 can successfully communicate with several different people who play the role of Tester #2. Next exercise: Present Tester #1 with a sequence of events. Tester #1 describes the sequence to Tester #2, who then uses the program and attempts to recreate the exact sequence that was seen by Tester #1. As before, Tester #2 gives what she thinks is the sequence back to Tester #1, who learns from the feedback.

- ***When the tester finds a defect that looks non-critical, she should run some additional tests to determine whether the observed failure is simply a weak symptom of a more serious problem.*** Consider the following exercise: At Florida Tech, we are constantly testing commercial software and so we have a population of student-found defects in our database. Before a student in my introductory testing class is allowed to enter new defect reports into the database, he must edit some old reports, performing the additional follow-up tests if they are needed, eventually submitting a clearer and more complete report. At present, I do the evaluation or a graduate student does it. We could also do peer evaluations.
- ***Testers must be able to spot ambiguities in requirements documents, specifications, etc.*** A group of my students and I are working through Spector (1997) as a first source of potential exercises.
- ***Failures can arise as a result of interaction of multiple variables. Testers must be able to plan efficient tests of interactions among variables.*** Exercise: Give the student a list of operating system versions, browser versions, printers, video cards, and communications devices (modems, network interface cards). The student creates a base set of configurations to test using the all pairs method (Cohen et al., 1997). Advanced version: Also provide the student with market data and customer support records that show that certain configurations are more popular or more prone to fail. The student is to make the smallest configuration test set that meets the all-pairs criterion and that also includes the key configurations. (Because the market-related configurations may involve the interaction of more than two critical variables, all-pairs doesn't automatically include these cases, and the student may end up with more than the all-pairs minimal set).
- ***Testers should be skilled at creating persuasive test cases that mimic real uses that involve naturally complex situations.*** There are a few different definitions of "scenario test." I use the term to mean a test that has four characteristics: (a) It is realistic, and reflects or is drawn from examples of actual tasks done by users. (b) It is complex, involving many features of the product. (c) It is easy for the tester to determine quickly whether the program passed or failed the test. And (d) At least one stakeholder who has power in the development decision process and who is outside of the test group would be interested and concerned if the program failed this test. Exercise: For a given application, design 5 scenario tests and explain how each one meets the 4 criteria.

I assert the following:

- we can identify many specific skills that are used by experienced software testers;
- we can find examples of the use of these skills (or the value of using them) in widely used software applications;
- the development of these skills by individual testers requires practice and feedback;
- for many of the skills, training in the form of carefully chosen "drill" exercises with sample answers will provide good enough education for many people; and
- for the other skills, additional feedback by a coach will be needed, but even these include opportunities for self-paced, private practice.

I propose to research and develop a skills-based approach to teaching many of the testing skills involving test planning, test design and execution, test result investigation and reporting. I expect that many of these can be taught at a distance, often through guided self-study. While people who successfully complete this training will not be expert testers, I believe that their skills will be recognized and valued by the test managers who supervise them.

For each skill, I expect to be able to identify or discover effective teaching methods:

1. That introduce the student to the skill. *(Often, this will be an explanation of the underlying concepts, an example of real-life application, and a step-by-step demonstration.)*
2. That support the student's first practice with the skill. *(Often, this will consist of a series of simple exercises.)*
3. That drill the student in the basic skill. *(Often, more exercises that are presented straightforwardly but are increasingly difficult.)*
4. That introduce the student to more complex problems that involve application of this skill. *(For example, rather than simply laying out a problem, show features of existing programs and point out cases in which a given technique is useful for testing a given feature.)*
5. That drill the student with challenging examples. *(For example, supply a program to the student and a list of several features. Have her identify the features that are candidates for using a given technique, and then lay out the application of that technique for each of the selected features.)*
6. That support expert application of the skill. *(For example, solving tough problems by combining the use of different techniques.)*

This is new ground. Books on software testing rarely include exercises, and even those that do include some (such as Myers, 1979, and Jorgensen, 1995) don't include many.

Validation

The list of skills will be circulated for review, especially among practitioners, because this list is an attempt to describe aspects of what they are doing.

Regarding effectiveness of the teaching methods, we will initially evaluate them by giving students problems and evaluating their solutions. Some of his work can be done in testing classes at Florida Tech. I will also evaluate modules with people who are not currently enrolled in my classes. These will include:

- University students who I recruit (and pay) to work through a skill module.
- Testing practitioners who participate on a voluntary basis.
- Other commercial and academic instructors who teach courses on software testing.

In each of these cases, I would work with before/after sets of problems, comparing the quality of answers given before training with the quality of answers to similar problems after training. Additionally, I'll collect subjective ratings of the effectiveness of the modules, at time of training and approximately three months after training.

None of the above measures will tell us whether the skill-based approach is better than competing methods. They will tell us whether the approach is substantially better than nothing (or than lecture-only teaching). As the materials become polished, and as other instructors suggest examples of better ways, we can run comparisons.

C.4 REUSABLE MATERIALS FOR EXERCISES AND TESTS

The skill-based approach described above requires development of many exercises and examples. I expect to make many of these broadly available in at least two ways:

- Publication of an outline / practice book (perhaps in the Schaum's series) or creation of a web site that presents the practice materials.

- Use of the materials in web-based or CD-based courses on software testing.

I plan to hold a subset of the exercises and examples in reserve, for use by teachers as test questions.

- These will be available on demand (sent by electronic mail to people who can convince me that they are testing course instructors) and/or will be included in a teacher's manual published alongside the Third Edition of *Testing Computer Software* (in preparation; the second edition is Kaner, et al., 1993, which John Wiley & Sons advertises as the best selling book--in terms of lifetime total sales--in the software testing field).

Over the three years, I will create a significant number of exercises for each skill module, many more than we'd need to demonstrate that the basic module is effective.

C.5 EXAMPLES OF SOFTWARE ERRORS

Appendix A of *Testing Computer Software* (Kaner et al., 1993) contains descriptions of over 480 types of software errors. Many additional failure reports are available on the web, at sites like bugnet.com, CNET.com, windowsannoyances.com and technical support sites of many software publishers. The RISKS forum also provides examples of often spectacular failures, as do leading computer magazines like ComputerWorld, eWeek, and InfoWorld.

These examples are useful but, based on my own teaching experience and the anecdotes of colleagues, they need work.

I propose to replicate many of these failures, creating screen shots and step-by-step walkthroughs that illustrate the problem. Here are some of the instructional uses:

- When a teacher demonstrates a testing technique, he can use real-life examples of defects and explain how a given technique would (or would not) have power (probability of finding a defect) against a defect of this type.
- Working from a subset of these examples, a teacher can show differences between testing techniques by explaining how (and why) one technique is more likely to find certain defects (demonstrate them) while the other is more likely to find others (demonstrate them).
- Scenarios (a series of screen shots, with commentary, that show a sequence of events that lead to a failure) are useful bases for homework exercises, in-class exercises, and examination questions (I've used them all three ways). Unfortunately, good scenarios are time-consuming to create. These examples will provide teachers with supporting material for many exercises.

I plan to post these examples on a website, assigning a permanent name and URL to each one. Authors and teachers will be able to distribute this material under the GPL free software license for documentation, enabling any teacher or author to use the examples freely. (Note: The authors and publisher of *Testing Computer Software* have already agreed to drop the Appendix in the Third Edition, so that derivative works of our second edition Appendix can go onto the web under GPL.)

A collection like this is valuable in its own right as an instructional tool but it also provides valuable additional benefits:

- Testers can audit their test plans by checking whether their planned testing of a given application would catch failures of the kinds that appear in this list. *Testing Computer Software* Appendix A is already used and useful for this, but it is dated, has serious gaps, and is often unclear to readers. It needs improvement.
- Testers and researchers can use the examples to evaluate and compare proposed software testing methods. See the discussions of Whittaker's fault model and of software testing paradigms below.

In software testing, we have very few shared examples. This web site has the potential of becoming a widely used resource.

Validation

We'll test these examples (run the steps and see the failures) before we post them on the web. The person who wrote the description won't be the person who follows its steps. Once a failure is posted, other people will attempt to use them and we will invite them to tell us about problems as and if they run into them.

C.6 OPEN SOURCE VERSIONS OF TESTING TOOLS

Several standard techniques are not routinely used in software publishing companies and are not routinely taught (or not taught in enough detail) in commercial software testing courses. A key reason for this is that these techniques require computation that is tedious and error prone if done by hand.

For example, consider the problem of compatibility testing (test this feature across operating systems, with different browsers, printers, mice, video cards, etc.) The number of possible test configurations is enormous. All-pairs combination testing is a straightforward method for selecting a relatively small number of multiple-variable test cases from a much larger population of possibilities (Cohen et al., 1997). Telcordia Technologies offers an extensive software service that will select the all-pairs (all-triples, all-quadruples, etc.) test set, even in massively complex situations. There is a brief free trial period, but after that, the service is not cheap. For teaching purposes, and for many practical purposes, we could happily use a less ambitious program that handles only 10 or 20 independent variables. I would like to fund a student to write such a program, and publish it as free software (open source, distributed under the GPL).

Similarly, we would benefit as teachers of testing methods if we had free software that did (or provided examples of):

- cause/effect graphing
- category partitioning
- orthogonal arrays
- table-driven automated tests for a few common GUI applications, showing how this is done under each of the main GUI regression test tools

and so on.

Validation

The measure of success of this work will be whether or not instructors are using the tools. We'll note whether we are using them at Florida Tech, and will solicit feedback from other teachers of software testing courses.

C.7 WHITTAKER'S FAULT MODEL AS A VEHICLE FOR TEACHING TECHNIQUES

Whittaker and Jorgensen (1999) argue that all software errors can be classified as failures to constrain inputs, outputs, storage, or computations. Based on that insight, they developed a collection of types of tests, which they call "attacks", that can expose instances of the different faults (Whittaker, 2000).

From an instructional and training point of view, Whittaker (2000) is a useful paper. The fault model is simple, but it can be used to generate a wide range of ideas for testing. The 20-plus attacks are well

explained and quickly lead students to productive testing in a laboratory setting. By "productive", I mean that students can use the list of techniques to find defects quickly. The attacks help the students organize their thinking and their approach to the software under test. This approach to organizing ideas about risks, errors, and tests has been used successfully as the underlying structure of some courses on software testing.

Even though the approach has obvious merits, there are open questions:

- How useful would these attacks have been for exposing errors during development that are now present in commercially sold software?
- What is the overlap among the attacks? Which types of errors are more likely to be found by which attacks?
- Are there examples of common errors that would *not* be routinely found in the normal course of use of any of the attacks?

Given a large and diverse collection of errors (see Section C.5 above), we can make progress toward answering these questions and, in the process we can strengthen this approach as a teaching structure for at least part of a course on software testing.

An Exploratory Experiment

This is a classification study, not a hypothesis-testing study.

For any given attack, there is a class of errors that will be routinely found in the normal use of the test. For example, imagine a program that expects all inputs to a specific numeric field to be in the range of 0 to 255, and has no defense against entries greater than 255. If you test every numeric input field by checking responses to out-of-bound input values, you will eventually find the error in this field in handling numbers greater than 255.

Given a large set of well-documented errors, we can sort the errors across attacks. Sorters will be experienced testers. They proceed with the error descriptions, one by one. For each error, the sorter decides whether any (one or more) of the attacks would routinely detect that error in the normal use of the attack. If so, we list that error with the associated attack(s).

Repeated across several testers, we will discover that:

- Specific types of errors are consistently judged likely to be found by certain attacks. *These are the ones that we use as examples when we teach the attacks.*
- Other errors are considered likely to be found, but there is disagreement as to how (which attack) the error will be found. *These suggest conceptual overlap among the attacks.*
- Finally, several errors are not consistently grouped with any attacks. *These suggest a need to develop new attacks.*

This is, of course, just an initial study. We can't be sure that a given attack *will* routinely detect a certain type of error until we check it with programs with known error characteristics. Those types of experiments will be conducted, but they are outside of the scope of the present proposal.[4] This initial study will provide a pattern that will be useful for teaching from, even if some of the details turn out to be wrong. It

4 Turky Al-Otaiby is a doctoral student at Florida Tech, and Alan Jorgensen is conducting post-doctoral research at Florida Tech. They are working independently. Both of them advise me that they plan to study the relationship between certain testing strategies or techniques and different types of errors.

will help us clarify our thinking and will serve as a strong pilot study for the more detailed and more costly research.

Validation Notes

It would be easy to run traditional hypothesis-testing statistics on this study's results. The obvious null hypothesis is that the attacks are unrelated to the defects and thus that in a set of independent matching trials of defects to *N* attacks, the probability that a given defect will be matched to a given attack will be 1/*N*. A chi-square test can be used to test such a hypothesis.

It would be astonishing if this null hypothesis was not rejected. The attacks were specifically designed to detect defects that occur frequently. For example, consider the following defect / attack pair:

- *The defect:* the program crashes in response to an input that is so huge that it overflows the program's input buffer.
- *The attack*: Enter extremely long strings into input fields, in order to try to overrun the input buffer.

If a tester did *not* sort this defect with this attack, we might wonder more about the tester than about the relationship between the defect and the attack. These obvious cases will cause any simple hypothesis of randomness to be rejected. That rejection won't teach us anything.

The more challenging objective is to extend the population of attacks, or improve the descriptions of attacks, in order to increase the number of immediately obvious defect/attack pairs. I don't have a valid statistical test to propose for this because we will be continuously increasing the set of analyzed defects. Any tests that look at proportions of defects covered will be confounded by selection biases. I think the more interesting measure is the simple list of defects for which there are directly related attacks. (One could also describe this as a list of risk / attack pairs.) That list of pairs is useful, even if it is incomplete.

Another point to recognize is that the set of errors analyzed is not a random sample, nor a representative sample of the population of possible errors in programs. This set will be the pool of errors that have come to our attention and that we have been able to replicate and describe. That process will leave behind an unknown number of other possible errors of an unknown number of types. The point is not to develop a population model that is amenable to statistical study. The point is to develop a set of risk / attack pairs that are intuitively obvious to experienced testers, useful for training inexperienced testers, and useful as reminders for all testers.

At some point, it will be useful to confirm experimentally (and/or through carefully observed case studies) that the attacks actually do help testers find the associated defects. That validation is beyond the scope of this project.

C.8 SOFTWARE TESTING PARADIGMS

Kaner and Bach (1999) described nine general black box testing styles or strategies, these being: domain testing, specification-based testing, risk-based testing, scenario-based testing, regression testing, exploratory testing, user-focused testing, single-function testing, and random testing (including stochastic tests based on state models).

These overlap. For example, one might use risk analysis to design a series of regression tests. Despite the logical overlap, our experience has been that some people focus on designing tests to mitigate a well researched set of risks, and then they choose to keep some of these tests for regression (retesting) purposes. Others focus more on the goal of reusability and less on the risks. The nine approaches captured

the ways that our clients described their approaches to us, or that we saw when we were retained as consultants to review and improve their testing processes.

As we consulted (independently of each other), we found that client companies usually focused on very few (three or fewer) of these approaches. As a result, they missed certain types of defects (or discovered them by luck) and suffered certain inefficiencies. By characterizing the testing strategies in these nine ways and explaining our characterizations, we were able to broaden the range of approaches that our clients used, making them (we believed) more effective as testing groups.

I've also found these strategies useful in testing classes, and they were very well received by attendees at the *Testing Computer Software* conference and at STAR West.

My main goals for this material are:

- Review the testing literature (academic and applied) to integrate these approaches with the classic descriptions of testing techniques and associated empirical results.
- Develop examples of tests that are prototypic of each of the styles of testing and select (from the pool we are developing, see Section C.5) examples of defects that would typically be found (or often missed) by a tester who used those types of tests.

Validation

In my commercial classes, this approach has been a useful structure for describing, contrasting, and comparing the material that I teach on software test design. It has also been useful for evaluating and describing the practices of my clients and James Bach's clients. With more rigor, better links to the researchers' and practitioners' literatures, and written worked examples, the approach *should be* more accessible and more useful for other teachers and other consultants. My measure of success of this work will be publication of a respectable paper or chapter that some teachers or consultants tell me they have adopted as a basis for some significant amount of their work.

C.9 WORKSHOPS ON TEACHING OF SOFTWARE TESTING

In 1997, I co-organized the first Los Altos Workshop on Software Testing (LAWST). The organizing question of the meeting was how to improve the maintainability of automated tests. The meeting differed from traditional conferences in a few ways. We focused on examples drawn from personal experience, presented solutions that either had worked or that had taught us something interesting when they failed. There were no time limits on the presentations. Some examples took a few minutes, others took longer. If you made a presentation, the other attendees could question you until they were finished with you. (In LAWST #5, we spent an entire day on one person's presentation.) We decided that it was more important to take the time we needed for each presentation than to run through the entire list of items offered by the attendees. The meetings were managed by a facilitator and a recorder. We produced no official minutes, but any attendee could publish anything presented or developed at the meeting. Kaner (1997) was the primary publication from the first meeting. For more details on the structure and agreements of the meetings, see the *LAWST Handbook* (Kaner & Lawrence, 1999).

LAWST was successful enough that we're now organizing the 12th meeting (June 2-3, 2001). The typical meeting is attended by 15-20 people (more than 20 is unmanageable in our format). Additionally, similar workshops have formed for other topics, such as the Software Test Managers Roundtable (which I facilitate), the Austin Workshop on Test Automation, the Virginia Workshop on Heuristic and Exploratory Techniques (which I facilitate), the Patterns of Software Testing workshop, and the Florida Workshop on Model-Based Testing (I'll facilitate its first meeting in February, 2001).

This meeting format is well suited to a discussion of teaching strategies and results. I plan to organize an annual workshop on the teaching of software testing, to be attended by a mix of university professors, in-house corporate teacher/trainers, and independent consultants who develop and teach their own courses.

Based on my experience with LAWST and its progeny, I expect that the people who attend these meetings will develop examples of their own, propose additions to the list of skills (and explain them), suggest new exercises, and generally add to the body of foundational material that this project will develop.

Validation

Feedback from the meeting participants, indicating whether the meetings are useful and how they can be improved. Also, the number of formal publications, conference talks, course collaborations, book collaborations, technical reports, and other outputs from the meetings.

C.10 WEB-BASED INSTRUCTION

A skill-based teaching approach should be well suited to distance learning. At Florida Tech, we are already planning to offer web-based courses on software testing. I don't think that a text-only format (no lectures at all) will serve this material well, nor am I enthusiastic about the lecture format that shows me (probably just my face) on part of the screen and my slides or my virtual whiteboard on the other part of the screen. We need demonstrations and planned, scripted (to address the issues likely to be in the minds of remote watchers) small group discussions. These will be expensive classes to develop. I believe that the expense will be justified by the higher educational value.

Most of the funding for the distance learning work will come from other grants that I will apply for later. The grant for this project will fund some early proofs of concept for distance learning classes. By creating a few segments from what will become the first few classes, I'll demonstrate the teaching style, obtain critical feedback, and lay a basis for evaluation of a focused distance learning funding proposal.

C.11 SCHEDULE OF DELIVERABLES

My dates are based on the assumption that funding begins in September 2001.

- ***Skills list, skills exercises, error examples:*** The laboratory will operate somewhat like a factory. It will take us a while to tool up, and experiment with our processes to become efficient. After that, we'll pump out new items on a continuing basis.
    - Open the website and publish the first skills list, exercises, and error examples—November 2001.
    - Either create a testing teachers' e-mail list or publicly invite teachers to the (invitation-only, moderated) software-testing list hosted by Brian Marick, James Bach, & Cem Kaner—November, 2001.
    - Update the website with new material—quarterly through September 2004 (and beyond if funding is extended)..
- ***Testing tools***
    - Specific order of development of tools depends on student interests.
    - Rate of development: two per year (First two complete by July, 2002).
- ***Evaluation and extension of Whittaker's fault model***

- Pilot study complete by January, 2003.
- First wave of sorts complete by September 2003.
- Second wave complete by May, 2004
- Publication of final results (tech report), September 2004.

- ***Testing paradigms***
  - Circulate strong first draft of a testing paradigms paper that includes extensive literature review—January 2002.
  - Circulate strong second draft that includes at least one detailed example for each paradigm/strategy—July, 2002.
  - Submit final draft for publication, with at least three detailed examples for each paradigm/strategy—January, 2003.
- ***Workshops***
  - Dates will be negotiated among the participants. Meetings will be held in Melbourne, FL. Because of our weather, I predict the meetings will be in late January, February, or early March. (one workshop in 2002, 2003 and in 2004)
- ***Distance learning***
  - First module, January 2002
  - Second and third (if there is a third) modules, September 2003

C.12 CLOSING NOTES ON THE PROPOSAL

The program solicitation specifies that the proposal will be evaluated on six dimensions. Here are some notes related to those criteria:

What is the Intellectual Merit of the Proposed Activity?

Software testing is an important area of work within information technology, but it has been largely ignored within the educational community. As a result, we face a shortage of appropriately educated people to do this work. The proposed activity will provide foundational material for teaching introductory and advanced classes on software testing. ***This kind of material is not available today. It will improve the quality of education in this field.*** The proposer has 18 years of industry experience, has written the best selling book on software testing, has developed commercially successful courses on software testing, has organized many workshops on software testing, has been active (as a regularly invited speaker or keynoter) on the software testing conference circuit for the last five years and is doing this work as a professor at a university that sees software testing research and teaching as one of the core competencies of its Computer Sciences department.

What are the Broader Impacts of the Proposed Activity?

The proposed activity is directly designed to promote teaching, training and learning--for example, through the development of the skills lists, the related exercises, the error examples, and the research on approaches to presenting test design in an organized and intuitively clear way. The activity advances discovery by creating common reference examples that will be useful to researchers, and by extending and deepening two frameworks for organizing thinking about software test design.

Software testing is an entry point into product development groups. In my experience, when members of underrepresented groups start their career in software development team, they often start in testing. The work proposed here is part of a broader project at Florida Tech to develop a set of educational materials for testers that will expand their technical expertise. Little advanced training in testing is available today—the advanced commercial courses typically focus on process management rather than technical growth. Based on my observations and discussions with other test managers and consultants, too many testers hit a dead end because so little technical training is available to them. If we improve training and educational opportunities for entry-level staff, we create opportunities for the underrepresented to earn promotions.

Additionally, the activity's focus on developing skill-based training lends itself to self-paced study and remote learning. As we develop this material, we should be able to develop first-rate courses that can be taught over the web, supported by meaningful self-paced exercises that come to the student with worked solutions (in the back of the book, or at the end of a link).

The proposed activity enhances the infrastructure for research and education by building a publicly available collection of examples, exercises, and tools. Additionally, part of the grant will support significant enhancement of the Florida Tech library collection of books related to testing, requirements analysis, usability analysis, empirical research design, law of software quality, and philosophy of science.

Dissemination of the results has been stressed throughout this proposal. Examples, exercises, and tools are published will be published on web sites and/or books, under terms that allow reuse (without requiring permission from the authors) by other teachers, trainers, students, or researchers. Other material will be published, in journals, books, workshops, or technical reports. The proposer has a track record of publishing a great deal of material on the web (see www.badsoftware.com and www.kaner.com), at conferences, workshops, and in journals and magazines.

The core societal benefit of this work lies in its high potential for improvement of software testing and through that, software quality. I've been an advocate for improving software quality throughout my career, as a user interface designer/developer, tester, software development manager, director, consultant, and attorney whose practice is focused on the law of software quality. The current trend in legislation and court decisions has been a sharp turning away from holding software companies accountable for defective products (s*ee, e.g.,* Kaner, 1999). While I continue to work to slow this trend (*see, e.g.,* Kaner & Pels, 2000; Koopman & Kaner, 2001), I concluded after careful consideration that I can do more to improve software quality by improving skill and professionalism within software testing than by any anything else that I can do. This conclusion led me to choose last year to accept a professorship, rather than continuing in a successful (and better paying) combination of computer law and software consulting practices.

Integration of Research and Education

The research proposed is all done in the service of improving education in software testing. The funding will largely support graduate and undergraduate students who will help me with the work. The project does not take away from my teaching duties (which are primarily in software testing and in computer law and ethics), and its funding for books will largely go for improvement of the library's collection (available for students as well as researchers) rather than my lab's private collection.

Integrating Diversity into NSF Programs, Projects, and Activities

See the notes above on broader impacts.

Innovation in Information Technology

See the notes above on intellectual merit.

C.13 LETTERS OF SUPPORT

I circulated a draft of this proposal and received several letters of support. Here are excerpts from some of them. I have tightened them up (to fit the limited space here) with the permission of the authors. No words have been added. Please feel free to contact me for the full originals.

From Sam Guckenheimer, Senior Director, Technology, Automated Testing Business Unit, Rational Software:

I am writing on behalf of Rational Software Corporation to express enthusiastic support for this proposal. This project will significantly advance the practices and knowledge resources available nationally for software quality professionals. We encourage you to fund this project.

Our qualifications*: Rational Software Corporation (RATL) is a major supplier of tools, services and training to support software development. Within Rational, I am responsible for furthering the development of best practices and products to support software testing. I am a regular speaker and participant nationally and internationally in conferences on software quality.*

What problem this project addresses*: Software testing accounts for approximately 50% of the cost and time of any major software project. Yet there are few standards and models available for training software testers, very few academic programs are available, and skilled professionals are in chronically short supply. This situation stands in marked contrast to software development, for which there are widely available curricula, degree programs at every major university, and many tens of thousands of new graduates annually. Where public software testing material exists, it has been subjected to little empirical validation and tends to be considerably at odds with current industry practice.*

What's special about this project: *This project can make an enormous contribution to the training of software testers. It will make a current, empirically validated curriculum publicly available for academia and industry. It will establish a base level of skills and knowledge for professional development and assessment. The training of best practices developed on Whittaker's "attacks" and Kaner & Bach's "paradigms" will be a major advance. It is refreshing to see a project draw on a wealth of practitioner experience and best practice drawn from across applicable industry.*

Qualifications of the PI: *Cem Kaner is uniquely suited to lead this work.He is one of the most innovative writers, speakers, and thinkers in the field of software test. He is the lead author of Testing Computer Software, widely considered the best book for training test practitioners. He has directly organized or indirectly spawned the vast majority of workshops to develop practitioner knowledge in the field. As professor at Florida Tech, he participates in the most developed software test training program available anywhere.*

From Ross Collard, Collard & Company, New York:

Our firm, Collard & Company, provides for-profit training in the areas of software testing and quality assurance to graduate-level working professionals. We have trained approximately 40,000 people in these topics over the last 15 years. You might say with a droll wit that we have lived comfortably over the years, because there is a huge shortfall in what the universities and professional associations are providing in these areas.

Our clients include most of the major U.S. organizations in high technology (Cisco, Dell, Intel, Microsoft, HP, etc.), most large financial institutions (American Express, Citigroup, Fed Reserve, etc.), industrial corporations (Ford, GE, etc.), and government (NASA, U.S. Air Force, etc.)

I also have taught seminars on these same topics for several universities, generally in the role of an adjunct professor. Educational institutions where I have conducted this training include the Assn. for Computing Machinery (ACM), the American Management Assn. (AMA), Boston University, George Washington University, Harvard University, IEEE, the International Quality & Productivity Centre (Australia), New York University, the Singapore Inst.of Management, Southern Methodist University, Stanford University, the University of California,Berkeley, the University of Colorado, the University of Michigan, and the University of Minnesota.

Virtually none of my university seminars have qualified for credit towards degrees, and virtually none have been at the undergraduate level.

The areas of software testing and qualify assurance are critically important to the U.S. and to the world.

There is a crying need to for the universities and professional associations to become serious about this training, develop academically qualified and effective materials, and incorporate it into their mainstream curricula.

I know the software testing field well, and I also want to say that if I had to select one person in the universe to undertake this critical mission, it would be you.

From Hans Buwalda, CMG TestFrame Research Center, Woerden, The Netherlands

My name is Hans Buwalda. It is a great pleasure and honor for me to write some words of support for the proposal of Cem Kaner. The kind of material he is proposing is very needed in the industry and it is a unique opportunity that somebody like Cem Kaner is willing to spend the time and energy needed to produce it.

First some words about myself. I work for CMG, a major European provider of ICT services. CMG is 12000 people large in numerous countries. Between 5 and 10 % of CMG's revenue is coming from testing, based on the method TestFrame, of which I'm the original developer and main architect since 1994. I have talked about the method and about other topics on testing and test management on several international conferences, among which are Quality Week and Star, the major conferences in the US.

I have read the proposal of Mr. Kaner. From our contacts with the academic world (which are quite extensive), I can confirm that the lack of good scientific work around testing is also felt in Europe and regarded here as an important problem. The need for skilled testers is hard to overestimate.

Testing is typically estimated at about 30% of all efforts in IT. This is about as much as programming for example. However, while there is a tremendous amount of attention for programming issues, testing is almost non-existent yet.

This initiative, especially with the name Cem Kaner behind it, could work as a badly needed catalyst to improve this situation.

From Brian Marick, Testing Foundations, Champaign, Illinois:

I am an independent consultant who teaches software testing to novice testers in commercial environments. I also taught testing to graduate students and advanced undergraduates for seven years as a part-time Visiting Lecturer at the University of Illinois. I am the author of The Craft of Software Testing.

I've read Professor Kaner's proposal and urge its support. To keep this note short, I'll single out two aspects that address a desperate problem.

But first, an apparent digression. The subject matter of programmers is programs. The good ones, especially, learn to program by immersing themselves in their subject matter. They find a program other people have worked on, and they wrestle with it. The program is the center of a web of artifacts (tools, languages, and documents) and practices (approaches and tricks of the trade). By changing the program, they learn the web.

In testing, this approach has been blocked. The subject matter of testers is bugs. But, whereas the subject matter of programmers is everywhere – the world is awash in good open source programs – good bugs are hard to find. It's not that there's no buggy software out there – quite the contrary – it's that the bugs can't be examined in detail the way that programs can be. They aren't suitable for learning testing. Professor Kaner's collection of examples will have a great impact.

Moreover, testing, like programming, is tool-dependent. But the tools are expensive and consequently scarce. A novice programmer can easily become proficient in free tools that are close approximations to the tools she will use in a commercial environment. That's not possible for testers. Professor Kaner's collection of classroom-level tools will help fill that gap.

This proposal will help unblock the tester education process and so is deserving of your support.

D. REFERENCES CITED

ACM/IEEE (1991) *Computing Curricula.*

ACM/IEEE (2001) *Computer Curricula.* (Draft March 6, 2000).

Cohen, D.M., S.R. Dalal, M.L. Fredman, & G.C. Patton,(1997) "The AETG System: An Approach to Testing Based on Combinatorial Design," *IEEE Transactions on Software Engineering,* Vol. 23 (27), July; www.argreenhouse.com/papers/gcp/AETGieee97.shtml.

Cusumano, M.A. & R. W. Selby (1995) *Microsoft Secrets*, Free Press.

Duchateau, P. & D. W. Zachmann (1986), *Schaum's Outline of Partial Differential Equations,* McGraw-Hill.

Dustin, E., J. Rashka, & J. Paul (1999) *Automated Software Testing: Introduction, Management, and Performance,* Addison-Wesley.

Fewster, M. & D. Graham (1999), *Software Test Automation : Effective Use of Test Execution Tools*, Addison-Wesley.

Gates, B. (1997) "Remarks by Bill Gates of Microsoft Corporation", *Gartner Symposium*, Orlando, FL; Originally available at www.microsoft.com/mscorp/billgates/billgates_l/speeches/gartner.htm; downloaded from www.oasis-open.org/cover/gates-gartnerXML.html.

Jorgensen, P.G. (1995), *Software Testing : A Craftsman's Approach*, CRC.

Kaner, C. (1997) "Improving the Maintainability of Automated Test Suites," *Software QA,* Volume 4, #4.

Kaner, C. (1999) "Software Engineering & UCITA." *John Marshall Journal of Computer & Information Law,* Volume 18, 2, pp. 435-546. Available at www.badsoftware.com/engr2000.htm

Kaner, C. (2000a) *Black Box Software Testing*, course notes, taught at UC Berkeley Extension, UC Santa Cruz Extension, and at dozens of corporate sites.

Kaner, C., (2000b), *Recruiting Software Testers*, workshop presented at several *Software Testing Analysis & Review (STAR) Conferences*, most recently *STAR WEST,* October 2000. Includes a 46 page draft chapter, "Recruiting Software Testers", to be published in Kaner, C., Bach, J., Nguyen, H., Johnson, B., Lawrence, B., & Falk, J. (in preparation), *Testing Computer Software* (3rd Ed.), Wiley.

Kaner, C. & J. Bach (1999) "Paradigms of Black Box Software Testing," *16th International Conference on Testing Computer Software*, Washington, D.C., June; Also, the keynote address at *Software Testing Analysis & Review Conference (STAR) WEST*, October, 1999, San Jose, CA.

Kaner, C., J. Falk, & H.Q. Nguyen (1993), *Testing Computer Software* (2nd Ed.), Wiley.

Kaner, C. & D. Hoffman (2000), *A Course on Software Test Automation*, course notes, taught at UC Santa Cruz Extension.

Kaner, C. & B. Lawrence (1999), *The LAWST Handbook*, distributed to meeting participants, available to others on request.

Kaner, C. & Pels, D. (1998), *Bad Software: What to Do When Software Fails.* Wiley.

Kaner, C. & Pels, D. (2000), "Comments Before the Federal Trade Commission in the Matter of High Technology Warranty Project FTC File No. P994413," available at www.badsoftware.com/ftc2000.htm.

Koopman, P. & Kaner, C. (2001), "The Problem of Embedded Software in UCITA and Drafts of Revised Article 2," *in press, UCC Bulletin.*

Myers, G.J. (1979), *The Art of Software Testing,* Wiley.

Pettichord, B. (2000), "Capture Replay: A Foolish Test Strategy", *Software Testing Analysis & Review Conference (STAR) WEST,* San Jose, CA..

Robinson, H. (2000), "Intelligent Test Automation", *Software Testing & Quality Engineering*, September / October; available at www.geocities.com/harry_robinson_testing/Intelligent_Test_Automation.htm.

Spector, C.C. (1997), *Saying One Thing, Meaning Another: Activities for Clarifying Ambiguous Language*, Thinking Publications.

Sveshnikov, A.A. & R. A. Silverman (Translator) (1979), *Problems in Probability Theory, Mathematical Statistics and Theory of Random Functions,* Dover.

Whittaker, J.A. (1997), "Stochastic software testing," *The Annals of Software Engineering*, Volume 4, pp. 115-131; available at www.geocities.com/harry_robinson_testing/whittaker97.doc

J. A. Whittaker (2000), "What is software testing. And why is it so hard," *IEEE Software*, Volume 17, 1 pp. 70-79.

Whittaker, J. A. (2000) "How to break software," *Software Testing Analysis & Review Conference (STAR) East*, Orlando, FL.

Whittaker, J. A. and A. A. Jorgensen, (1999) "Why Software Fails." *ACM Software Engineering Notes*, Volume 24 (4). Also awarded "Best Presentation" at *Software Testing Analysis & Review Conference (STAR) EAST*, 1999, Orlando, FL.

E. BIOGRAPHICAL SKETCH

a. Professional Preparation

- Brock University, Ontario — Arts & Sciences, primarily Math & Philosophy — B.A., 1974
- McMaster University, Ontario — Experimental Psychology — Ph.D. 1984
- Golden Gate University, CA — Law — J.D. 1994

b. Appointments

- Professor of Computer Sciences, Florida Institute of Technology, August 2000 – present
- Extension Instructor, University of California Extension (Berkeley and Santa Cruz), 1994-2000
- Attorney, Law Office of Cem Kaner, 1994-present [My client list is private, but it doesn't include anyone relevant to this grant, other than people who are elsewhere listed as collaborators].
- Proprietor, kaner.com (software consulting firm), 1993-present [My clients have included Avid Technologies, Aveo, Barra, BEA Systems, BMC, Broderbund Software, Catalysis (through them, the California Dept of Transportation), CDI, Cognos, Compaq, Fenwick & West, Fonix, Gilbarco, Hammer Technologies, Hewlett-Packard, IDTS, Intel, Iomega, Kodak, Metamor (Vanteon, Turning Point Software), Microsoft, MyTurn.com, New Paradigms, Oracle / Network Computer (now Liberate Technologies), OrCAD, Parametric Technologies, Peoplesoft, Postalsoft, PowerQuest, Quarterdeck, Reliable Software Technologies (now Cigital), Safeco, ShareData (now E-Trade), Software AG, Software Quality Engineering, Software Test Labs (now Data Dimensions), Stevedoring Services of America, Symantec, Testing Testing 123, the WELL, and Wind River]
- Deputy District Attorney (full-time volunteer), Santa Clara County, April 1994-July 1994.
- Law Clerk, Law Office of Berne Reuben, Dec. 1993-April 1994.
- Director of Documentation and Software Testing; Documentation Group Manager; Software Development Manager, Power Up Software (later Spinnaker Software), 1989-1994.
- Software Test Manager, Electronic Arts, 1988
- Human Factors Analyst / Software Engineer, Telenova, Inc, 1984-1988.
- Testing Technology Team Leader; Software Testing Supervisor, MicroPro (WordStar), 1983-1984.
- Associate (and then Senior Associate) (part-time), Psylomar Organization Development, 1983-1985.
- Owner trainee (database analyst, inventory systems analyst, store manager, salesperson, warehouse clerk, cashier) (full-time, part time, various arrangements) Kaners, Inc., and 1+1 Inc., 1966-1984.

c. Publications

c.(i) Most closely related

- Kaner, C., J. Falk, & H.Q. Nguyen (1993), Testing Computer Software (2nd Ed.), Wiley.
- Kaner, C. (2000) *Black Box Software Testing*, course notes, taught at UC Berkeley Extension, UC Santa Cruz Extension, and at dozens of corporate sites.
- Kaner, C. & J. Bach (1999) "Paradigms of Black Box Software Testing," *16th International Conference on Testing Computer Software*, Washington, D.C., June; Also, the keynote address at *Software Testing Analysis & Review Conference (STAR) WEST*, October, 1999, San Jose, CA.
- Kaner, C. (2000). Recruiting Software Testers, (paper & 1-day seminar) *Software Testing Analysis & Review Conference (STAR) East*, Orlando, FL.

- Kaner, C. (2000). An Outline for Software Testing Outsourcing, *Software Testing Analysis & Review Conference (STAR) East,* Orlando, FL.

c.(ii) Other significant publications

- Kaner, C. & Pels, D. (1998), *Bad Software: What to Do When Software Fails.* Wiley.
- Kaner, C. (2000). Architectures of Test Automation, *Software Testing Analysis & Review Conference (STAR) West*, San Jose, CA.
- Kaner, C. (1999) "Software Engineering & UCITA." *John Marshall Journal of Computer & Information Law*, Volume 18, 2, pp. 435-546. Available at www.badsoftware.com/engr2000.htm
- Kaner, C. (2000) "Measurement of the Extent of Testing," *Pacific Northwest Software Quality Conference*, Portland, OR.
- Marick, B., J. Bach, & C. Kaner (2000). A Manager's Guide to Evaluating Test Suites, *International Software Quality Week,* San Francisco, CA. See http://www.testing.com/writings.html

d. Synergistic Activities

- **Co-Organizer of several ongoing workshops in software testing**, including the Los Altos Workshops on Software Testing, the Software Test Managers Roundtable, the Workshop on Heuristic and Exploratory Techniques, and the Florida Workshops on Model-Based Software Test Automation.
- **Elected to the American Law Institute, May, 1999.** One of the most prestigious organizations of attorneys in the country, memberships are life-long and the Institute's total membership is limited to 3000. This organization drafts legislation, treaties, and the Restatements (of Torts, Products Liability, Agency, Contracts, etc.), a series of authoritative treaties that are heavily relied on by appellate judges. I was elected in recognition of my legislative work on computer law.
- **Developing a curriculum for a software testing degree.** I've been sketching this out and circulating informal drafts for a year. I'm scheduled to present a first "formal" public proposal at the Software Testing Analysis and Review Conference this May (2001).

e. Collaborators and Other Affiliations

e.(i) Collaborators: James Bach (Satisfice Inc), Jim Bampos, Shirley Becker (Florida Tech), Jean Braucher (University of Arizona), Jennifer Brock (ADP), Hans Buwalda (CMG), Ross Collard (independent), Jack Falk (Blue Martini Software), Ed Foster (InfoWorld), David Gelperin (SQE), Jens Gregor (U. Tennessee), Sam Guckenheimer (Rational) Payson Hall (Catalysis Group), Elisabeth Hendrickson (Quality Tree Consulting), Gail Hillebrand (Consumers Union), Doug Hoffman (Blue Martini Software), Bob Johnson (Agorics), Mark Johnson (Cadence), Alan Jorgensen (Florida Tech), Phil Koopman (Carnegie Mellon University), Brian Lawrence (Coyote Valley Software), James Love (Consumer Project on Technology), Brian Marick (University of Illinois, Urbana-Champaign), Gerald Marin (U. Central Florida), Fran McKain (Hewlett-Packard), David McMahon (attorney), Renaldo Menezes (Florida Tech), Mark Minasi (independent journalist), Hung Quoc Nguyen (LogiGear Technologies), Noel Nyman (Microsoft), David Pels (Snap-on Tools), Brett Pettichord (pettichord.com), Neal Reizer (UC Extension), David Rice (University of Rhodes Island, Law School), Sharon Roberts (Roberts consulting; Independent Computer Consultants Association), Johanna Rothman (Johanna Rothman Consulting), Melora Svoboda, Michael Thomason (U. Tennessee), James Tierney (Microsoft), Clark Savage Turner (Cal Poly), James Whittaker (Florida Tech).

e.(ii) Graduate and Post Doctoral Advisors. A.B. Kristofferson (retired), John R. Platt (McMaster U.), Woody Heron (retired), Ibrahim Ahmad (Mathematics, McMaster U.), Roger Bernhardt (Golden Gate U. Law School).

e.(iii) Thesis Advisor and Postgraduate-Scholar Sponsor: Ibrahim El-Far, Amit Singh, Turky Al-Otaiby (all at Florida Tech.).

SUMMARY PROPOSAL BUDGET

YEAR 1

FOR NSF USE ONLY		
PROPOSAL NO.	DURATION (months) Proposed	Granted
AWARD NO.		

ORGANIZATION: **Florida Institute of Technology**

PRINCIPAL INVESTIGATOR / PROJECT DIRECTOR: **Cem Kaner**

A. SENIOR PERSONNEL: PI/PD, Co-PI's, Faculty and Other Senior Associates (List each separately with title, A.7. show number in brackets)	NSF Funded Person-mos. CAL	ACAD	SUMR	Funds Requested By proposer	Funds granted by NSF (if different)
1. **Cem Kaner - Professor**	**1.50**	**0.00**	**1.50**	$ **20,000**	$
2.					
3.					
4.					
5.					
6. (**0**) OTHERS (LIST INDIVIDUALLY ON BUDGET JUSTIFICATION PAGE)	**0.00**	**0.00**	**0.00**	**0**	
7. (**1**) TOTAL SENIOR PERSONNEL (1 - 6)	**1.50**	**0.00**	**1.50**	**20,000**	
B. OTHER PERSONNEL (SHOW NUMBERS IN BRACKETS)					
1. (**0**) POST DOCTORAL ASSOCIATES	**0.00**	**0.00**	**0.00**	**0**	
2. (**0**) OTHER PROFESSIONALS (TECHNICIAN, PROGRAMMER, ETC.)	**0.00**	**0.00**	**0.00**	**0**	
3. (**2**) GRADUATE STUDENTS				**45,000**	
4. (**2**) UNDERGRADUATE STUDENTS				**35,000**	
5. (**0**) SECRETARIAL - CLERICAL (IF CHARGED DIRECTLY)				**0**	
6. (**150**) OTHER				**6,000**	
TOTAL SALARIES AND WAGES (A + B)				**106,000**	
C. FRINGE BENEFITS (IF CHARGED AS DIRECT COSTS)				**4,600**	
TOTAL SALARIES, WAGES AND FRINGE BENEFITS (A + B + C)				**110,600**	
D. EQUIPMENT (LIST ITEM AND DOLLAR AMOUNT FOR EACH ITEM EXCEEDING $5,000.)					
TOTAL EQUIPMENT				**0**	
E. TRAVEL 1. DOMESTIC (INCL. CANADA, MEXICO AND U.S. POSSESSIONS)				**5,000**	
2. FOREIGN				**0**	
F. PARTICIPANT SUPPORT COSTS					
1. STIPENDS $ **0**					
2. TRAVEL **7,500**					
3. SUBSISTENCE **0**					
4. OTHER **0**					
TOTAL NUMBER OF PARTICIPANTS (**15**) TOTAL PARTICIPANT COSTS				**7,500**	
G. OTHER DIRECT COSTS					
1. MATERIALS AND SUPPLIES				**11,000**	
2. PUBLICATION COSTS/DOCUMENTATION/DISSEMINATION				**950**	
3. CONSULTANT SERVICES				**0**	
4. COMPUTER SERVICES				**0**	
5. SUBAWARDS				**0**	
6. OTHER				**5,000**	
TOTAL OTHER DIRECT COSTS				**16,950**	
H. TOTAL DIRECT COSTS (A THROUGH G)				**140,050**	
I. INDIRECT COSTS (F&A)(SPECIFY RATE AND BASE) **Indirect costs (Rate: 11.0000, Base: 150050)**					
TOTAL INDIRECT COSTS (F&A)				**16,506**	
J. TOTAL DIRECT AND INDIRECT COSTS (H + I)				**156,556**	
K. RESIDUAL FUNDS (IF FOR FURTHER SUPPORT OF CURRENT PROJECTS SEE GPG II.D.7.j.)				**0**	
L. AMOUNT OF THIS REQUEST (J) OR (J MINUS K)				**156,556**	

M. COST SHARING PROPOSED LEVEL $ **0** AGREED LEVEL IF DIFFERENT $

PI / PD TYPED NAME & SIGNATURE*: **Cem Kaner** DATE

ORG. REP. TYPED NAME & SIGNATURE* DATE

FOR NSF USE ONLY		
INDIRECT COST RATE VERIFICATION		
Date Checked	Date Of Rate Sheet	Initials - ORG

SUMMARY PROPOSAL BUDGET

YEAR 2

FOR NSF USE ONLY		
PROPOSAL NO.	DURATION (months) Proposed	Granted
AWARD NO.		

ORGANIZATION: **Florida Institute of Technology**

PRINCIPAL INVESTIGATOR / PROJECT DIRECTOR: **Cem Kaner**

A. SENIOR PERSONNEL: PI/PD, Co-PI's, Faculty and Other Senior Associates (List each separately with title, A.7. show number in brackets)	NSF Funded Person-mos. CAL	ACAD	SUMR	Funds Requested By proposer	Funds granted by NSF (if different)
1. **Cem Kaner - Professor**	**1.50**	**0.00**	**1.50**	$ **20,000**	$
2.					
3.					
4.					
5.					
6. (**0**) OTHERS (LIST INDIVIDUALLY ON BUDGET JUSTIFICATION PAGE)	**0.00**	**0.00**	**0.00**	**0**	
7. (**1**) TOTAL SENIOR PERSONNEL (1 - 6)	**1.50**	**0.00**	**1.50**	**20,000**	
B. OTHER PERSONNEL (SHOW NUMBERS IN BRACKETS)					
1. (**0**) POST DOCTORAL ASSOCIATES	**0.00**	**0.00**	**0.00**	**0**	
2. (**0**) OTHER PROFESSIONALS (TECHNICIAN, PROGRAMMER, ETC.)	**0.00**	**0.00**	**0.00**	**0**	
3. (**2**) GRADUATE STUDENTS				**45,000**	
4. (**2**) UNDERGRADUATE STUDENTS				**35,000**	
5. (**0**) SECRETARIAL - CLERICAL (IF CHARGED DIRECTLY)				**0**	
6. (**150**) OTHER				**6,000**	
TOTAL SALARIES AND WAGES (A + B)				**106,000**	
C. FRINGE BENEFITS (IF CHARGED AS DIRECT COSTS)				**4,600**	
TOTAL SALARIES, WAGES AND FRINGE BENEFITS (A + B + C)				**110,600**	
D. EQUIPMENT (LIST ITEM AND DOLLAR AMOUNT FOR EACH ITEM EXCEEDING $5,000.)					
TOTAL EQUIPMENT				**0**	
E. TRAVEL 1. DOMESTIC (INCL. CANADA, MEXICO AND U.S. POSSESSIONS)				**5,000**	
2. FOREIGN				**0**	
F. PARTICIPANT SUPPORT COSTS					
1. STIPENDS $ **0**					
2. TRAVEL **7,500**					
3. SUBSISTENCE **0**					
4. OTHER **0**					
TOTAL NUMBER OF PARTICIPANTS (**15**) TOTAL PARTICIPANT COSTS				**7,500**	
G. OTHER DIRECT COSTS					
1. MATERIALS AND SUPPLIES				**11,000**	
2. PUBLICATION COSTS/DOCUMENTATION/DISSEMINATION				**950**	
3. CONSULTANT SERVICES				**0**	
4. COMPUTER SERVICES				**0**	
5. SUBAWARDS				**0**	
6. OTHER				**5,000**	
TOTAL OTHER DIRECT COSTS				**16,950**	
H. TOTAL DIRECT COSTS (A THROUGH G)				**140,050**	
I. INDIRECT COSTS (F&A)(SPECIFY RATE AND BASE) **Indirect costs (Rate: 11.0000, Base: 150050)**					
TOTAL INDIRECT COSTS (F&A)				**16,506**	
J. TOTAL DIRECT AND INDIRECT COSTS (H + I)				**156,556**	
K. RESIDUAL FUNDS (IF FOR FURTHER SUPPORT OF CURRENT PROJECTS SEE GPG II.D.7.j.)				**0**	
L. AMOUNT OF THIS REQUEST (J) OR (J MINUS K)				$ **156,556**	

M. COST SHARING PROPOSED LEVEL $ **0** AGREED LEVEL IF DIFFERENT $

PI / PD TYPED NAME & SIGNATURE*	DATE	FOR NSF USE ONLY
Cem Kaner		INDIRECT COST RATE VERIFICATION
ORG. REP. TYPED NAME & SIGNATURE*	DATE	Date Checked / Date Of Rate Sheet / Initials - ORG

SUMMARY PROPOSAL BUDGET

YEAR 3

FOR NSF USE ONLY		
PROPOSAL NO.	DURATION (months) Proposed	Granted
AWARD NO.		

ORGANIZATION: **Florida Institute of Technology**

PRINCIPAL INVESTIGATOR / PROJECT DIRECTOR: **Cem Kaner**

A. SENIOR PERSONNEL: PI/PD, Co-PI's, Faculty and Other Senior Associates (List each separately with title, A.7. show number in brackets)	NSF Funded Person-mos. CAL	ACAD	SUMR	Funds Requested By proposer	Funds granted by NSF (if different)
1. **Cem Kaner - Professor**	**1.50**	**0.00**	**1.50**	$ **20,000**	$
2.					
3.					
4.					
5.					
6. (**0**) OTHERS (LIST INDIVIDUALLY ON BUDGET JUSTIFICATION PAGE)	**0.00**	**0.00**	**0.00**	**0**	
7. (**1**) TOTAL SENIOR PERSONNEL (1 - 6)	**1.50**	**0.00**	**1.50**	**20,000**	
B. OTHER PERSONNEL (SHOW NUMBERS IN BRACKETS)					
1. (**0**) POST DOCTORAL ASSOCIATES	**0.00**	**0.00**	**0.00**	**0**	
2. (**0**) OTHER PROFESSIONALS (TECHNICIAN, PROGRAMMER, ETC.)	**0.00**	**0.00**	**0.00**	**0**	
3. (**2**) GRADUATE STUDENTS				**45,000**	
4. (**2**) UNDERGRADUATE STUDENTS				**35,000**	
5. (**0**) SECRETARIAL - CLERICAL (IF CHARGED DIRECTLY)				**0**	
6. (**150**) OTHER				**6,000**	
TOTAL SALARIES AND WAGES (A + B)				**106,000**	
C. FRINGE BENEFITS (IF CHARGED AS DIRECT COSTS)				**4,600**	
TOTAL SALARIES, WAGES AND FRINGE BENEFITS (A + B + C)				**110,600**	
D. EQUIPMENT (LIST ITEM AND DOLLAR AMOUNT FOR EACH ITEM EXCEEDING $5,000.)					
TOTAL EQUIPMENT				**0**	
E. TRAVEL 1. DOMESTIC (INCL. CANADA, MEXICO AND U.S. POSSESSIONS)				**5,000**	
2. FOREIGN				**0**	
F. PARTICIPANT SUPPORT COSTS					
1. STIPENDS $ **0**					
2. TRAVEL **7,500**					
3. SUBSISTENCE **0**					
4. OTHER **0**					
TOTAL NUMBER OF PARTICIPANTS (**15**) TOTAL PARTICIPANT COSTS				**7,500**	
G. OTHER DIRECT COSTS					
1. MATERIALS AND SUPPLIES				**11,000**	
2. PUBLICATION COSTS/DOCUMENTATION/DISSEMINATION				**950**	
3. CONSULTANT SERVICES				**0**	
4. COMPUTER SERVICES				**0**	
5. SUBAWARDS				**0**	
6. OTHER				**5,000**	
TOTAL OTHER DIRECT COSTS				**16,950**	
H. TOTAL DIRECT COSTS (A THROUGH G)				**140,050**	
I. INDIRECT COSTS (F&A)(SPECIFY RATE AND BASE) **Indirect costs (Rate: 11.0000, Base: 150050)**					
TOTAL INDIRECT COSTS (F&A)				**16,506**	
J. TOTAL DIRECT AND INDIRECT COSTS (H + I)				**156,556**	
K. RESIDUAL FUNDS (IF FOR FURTHER SUPPORT OF CURRENT PROJECTS SEE GPG II.D.7.j.)				**0**	
L. AMOUNT OF THIS REQUEST (J) OR (J MINUS K)				$ **156,556**	

M. COST SHARING PROPOSED LEVEL $ **0** AGREED LEVEL IF DIFFERENT $

PI / PD TYPED NAME & SIGNATURE*	DATE	FOR NSF USE ONLY
Cem Kaner		INDIRECT COST RATE VERIFICATION
ORG. REP. TYPED NAME & SIGNATURE*	DATE	Date Checked / Date Of Rate Sheet / Initials - ORG

SUMMARY PROPOSAL BUDGET

Cumulative

FOR NSF USE ONLY		
PROPOSAL NO.	DURATION (months) Proposed	Granted
AWARD NO.		

ORGANIZATION: **Florida Institute of Technology**

PRINCIPAL INVESTIGATOR / PROJECT DIRECTOR: **Cem Kaner**

A. SENIOR PERSONNEL: PI/PD, Co-PI's, Faculty and Other Senior Associates (List each separately with title, A.7. show number in brackets)	NSF Funded Person-mos. CAL	ACAD	SUMR	Funds Requested By proposer	Funds granted by NSF (if different)
1. **Cem Kaner - Professor**	**4.50**	**0.00**	**4.50**	$ **60,000**	$
2.					
3.					
4.					
5.					
6. ( ) OTHERS (LIST INDIVIDUALLY ON BUDGET JUSTIFICATION PAGE)	**0.00**	**0.00**	**0.00**	**0**	
7. (**1**) TOTAL SENIOR PERSONNEL (1 - 6)	**4.50**	**0.00**	**4.50**	**60,000**	
B. OTHER PERSONNEL (SHOW NUMBERS IN BRACKETS)					
1. (**0**) POST DOCTORAL ASSOCIATES	**0.00**	**0.00**	**0.00**	**0**	
2. (**0**) OTHER PROFESSIONALS (TECHNICIAN, PROGRAMMER, ETC.)	**0.00**	**0.00**	**0.00**	**0**	
3. (**6**) GRADUATE STUDENTS				**135,000**	
4. (**6**) UNDERGRADUATE STUDENTS				**105,000**	
5. (**0**) SECRETARIAL - CLERICAL (IF CHARGED DIRECTLY)				**0**	
6. (**450**) OTHER				**18,000**	
TOTAL SALARIES AND WAGES (A + B)				**318,000**	
C. FRINGE BENEFITS (IF CHARGED AS DIRECT COSTS)				**13,800**	
TOTAL SALARIES, WAGES AND FRINGE BENEFITS (A + B + C)				**331,800**	
D. EQUIPMENT (LIST ITEM AND DOLLAR AMOUNT FOR EACH ITEM EXCEEDING $5,000.)					
TOTAL EQUIPMENT				**0**	
E. TRAVEL 1. DOMESTIC (INCL. CANADA, MEXICO AND U.S. POSSESSIONS)				**15,000**	
2. FOREIGN				**0**	
F. PARTICIPANT SUPPORT COSTS					
1. STIPENDS $ **0**					
2. TRAVEL **22,500**					
3. SUBSISTENCE **0**					
4. OTHER **0**					
TOTAL NUMBER OF PARTICIPANTS (**45**) TOTAL PARTICIPANT COSTS				**22,500**	
G. OTHER DIRECT COSTS					
1. MATERIALS AND SUPPLIES				**33,000**	
2. PUBLICATION COSTS/DOCUMENTATION/DISSEMINATION				**2,850**	
3. CONSULTANT SERVICES				**0**	
4. COMPUTER SERVICES				**0**	
5. SUBAWARDS				**0**	
6. OTHER				**15,000**	
TOTAL OTHER DIRECT COSTS				**50,850**	
H. TOTAL DIRECT COSTS (A THROUGH G)				**420,150**	
I. INDIRECT COSTS (F&A)(SPECIFY RATE AND BASE)					
TOTAL INDIRECT COSTS (F&A)				**49,518**	
J. TOTAL DIRECT AND INDIRECT COSTS (H + I)				**469,668**	
K. RESIDUAL FUNDS (IF FOR FURTHER SUPPORT OF CURRENT PROJECTS SEE GPG II.D.7.j.)				**0**	
L. AMOUNT OF THIS REQUEST (J) OR (J MINUS K)				$ **469,668**	

M. COST SHARING PROPOSED LEVEL $ **0** AGREED LEVEL IF DIFFERENT $

PI / PD TYPED NAME & SIGNATURE*: **Cem Kaner** DATE

ORG. REP. TYPED NAME & SIGNATURE* DATE

FOR NSF USE ONLY

INDIRECT COST RATE VERIFICATION

Date Checked	Date Of Rate Sheet	Initials - ORG

NSF Form 1030 (10/99) Supersedes all previous editions

C***SIGNATURES REQUIRED ONLY FOR REVISED BUDGET (GPG III.B)**

Budget Justification Page

The budget justification is the same for each of the three years:

A. Senior Personnel-- Cem Kaner's current 9 month salary is $125,000 ($13,889 per month). At the same rate, this works out to $20833 for 1.5 months. The budgeted amount is $20,000 for 1.5 months. This will stay flat over the three years, even though Kaner's salary will probably increase in August 2001, 2002, and 2003.

B. Other Personnel

Fully supported graduates and undergraduates will receive approximately $30,000 per 9-month year, including tuition. However, some students have scholarships or will also work as teaching assistants. And until I know them well, I will pay Masters students and undergraduates by the hour, without tuition support. Therefore, I may be able to support more than the four students that appear in this breakdown. In that event, we will be able to create more exercises and examples per year and perhaps be able to build another tool per year.

The 150 "Other" personnel are students who participate in experiments. The estimate of 200 people is an estimated total across the three years. Many of these people will participate in more than one experiment (repeaters will be trained in more than one skill). The skill development experiments (or individual trials) will take less time per unit than the sorting experiment, and will not require as much prior familiarity with the software testing field.

Over three years, we will probably do three sorting experiments, with an estimated average of 10 students per year (fewer in academic 2001/2002, more later), at an estimated average of 20 hours per student. At $20 per hour (the going rate for the better senior students here), this is $400 per year per student, for a total of $4000. Over the three years, I hope to do fifteen skills experiments (an average of 5 per year). These will probably take 3 hours each. At a rate of $50 per student (totalled over the 3 hours), $2000 pays for 40 student-sessions, or 8 students per experiment. Some tests will require more students than this, but in some cases, we will be able to add volunteers.

D. EQUIPMENT

The lab will buy several pieces of equipment, such as a digital camcorder, LCD projectors, at least two portable computers, second monitors (so that we can run the software in debug mode and see the running program and the debug messages in parallel). This equipment will make it easy for us to watch a senior tester actually working and to record her actions for future study. At a recent Workshop on Heuristic & Exploratory Techniques, James Bach gave a powerful demonstration of the value of having several people watch a projected screen while one person actually ran the tests. We are trying to discover what skills are used by senior testers and then trying to decide how to train people in them. We will make more discoveries, more accurately, and faster if we watch senior testers at work. I expect that some of the computer equipment will be used as the primary work computer(s) of the students who help me conduct this research.

None of these pieces of equipment will cost $5000. We expect to spend a total of about $10000 per year on equipment, and have included that $10000 estimate in G. MATERIALS AND SUPPLIES, below.

Budget Justification Page

E. TRAVEL

The $5000 pays for me, and/or a student, to travel to conferences and present our findings. I speak at many conferences, and will present the findings of this research several times over the three years. The $5000 number is so low because many of them are willing to cover conference fee and to subsidize my travel.

F. PARTICIPANT SUPPORT COSTS

We will host a workshop on the teaching of software testing in the winters of 2001/2002, 2002/2003, and 2003/2004. Based on my experiences at the Los Altos Workshops on Software Testing, the Software Test Managers Roundtable, the Austin Workshop on Test Automation, and the Workshop on Heuristic & Exploratory Techniques, the best size for the workshop is 13-18 participants. $7500 allows us to subsidize travel to the workshop (airfare and overnight lodging) at a level of approximately $500 per person.

G. OTHER DIRECT COSTS

G.1 MATERIALS AND SUPPLIES As mentioned above (equipment), about $10,000 per year will be needed for (non-capital) equipment. We expect to spend another $1000 per year on supplies.

G.2 PUBLICATION COSTS / DOCUMENTATION / DISTRIBUTION. I estimate a cost of about $600 per year for access and storage on a commercial web service provider. Our local server, www.se.fit.edu, is getting overburdened. Until we solve that problem, we'll use a service like Earthlink to host the web site that posts the results of this project (primarily the exercises and the examples). The remaining $350 is for telephone, postage, and the cost of the domain name.

G.6 OTHER covers $5000 for books and subscriptions. Florida Tech's library is seriously inadequate in books and journals involving testing, quality control, software usability, software reliability, etc. We will buy materials that will directly assist us in this project's research, but make the University library the permanent home of the collection. This will give our other testing-interested graduate and undergraduate students access to a more adequate set of materials.

Current and Pending Support

(See GPG Section II.D.8 for guidance on information to include on this form.)

The following information should be provided for each investigator and other senior personnel. Failure to provide this information may delay consideration of this proposal.

Investigator: **Cem Kaner**

Other agencies (including NSF) to which this proposal has been/will be submitted.

Support: ☒ Current ☐ Pending ☐ Submission Planned in Near Future ☐ *Transfer of Support

Project/Proposal Title: **ITR/SY+PE: IMPROVING THE EDUCATION OF SOFTWARE TESTERS**

Source of Support: **There is no other support for this project**

Total Award Amount: $ **0** Total Award Period Covered: **01/01/00 - 01/01/00**

Location of Project: **Florida Tech, Department of Computer Sciences**

Person-Months Per Year Committed to the Project. Cal: **37.50** Acad: **36.00** Sumr: **1.50**

Support: ☐ Current ☐ Pending ☐ Submission Planned in Near Future ☐ *Transfer of Support

Project/Proposal Title:

Source of Support:

Total Award Amount: $ Total Award Period Covered:

Location of Project:

Person-Months Per Year Committed to the Project. Cal: Acad: Sumr:

Support: ☐ Current ☐ Pending ☐ Submission Planned in Near Future ☐ *Transfer of Support

Project/Proposal Title:

Source of Support:

Total Award Amount: $ Total Award Period Covered:

Location of Project:

Person-Months Per Year Committed to the Project. Cal: Acad: Sumr:

Support: ☐ Current ☐ Pending ☐ Submission Planned in Near Future ☐ *Transfer of Support

Project/Proposal Title:

Source of Support:

Total Award Amount: $ Total Award Period Covered:

Location of Project:

Person-Months Per Year Committed to the Project. Cal: Acad: Sumr:

Support: ☐ Current ☐ Pending ☐ Submission Planned in Near Future ☐ *Transfer of Support

Project/Proposal Title:

Source of Support:

Total Award Amount: $ Total Award Period Covered:

Location of Project:

Person-Months Per Year Committed to the Project. Cal: Acad: Summ:

*If this project has previously been funded by another agency, please list and furnish information for immediately preceding funding period.

H. FACILITIES, EQUIPMENT AND OTHER RESOURCES

Florida Tech's primary resources in support of this project are Professor Shirley Becker, Associate Professor James Whittaker, Adjunct Professor Alan Jorgensen, several graduate students, and me. We are a strong group of teachers who have a deep interest in software testing issues.

Florida Tech has a Software Engineering Lab with a variety of PC-compatible computers, a second lab for graduate assistants (with PC's), and a third lab with UNIX systems. These will meet my main equipment needs. We will probably add two multimedia systems to this that contain:

- a portable computer equipped for multimedia support
- digital camcorder
- digital projector
- second monitor so that we can see debug mode output on the second monitor in parallel with the main application running
- CD-burner

Setups like this are very desirable when you have 3-4 people sitting in a lab together trying to figure out as a group how to replicate or troubleshoot a problem. The camcorders will also be useful for capturing interesting content for distance learning segments.

Administrative support services will be provided by the Department (this project will not add substantially to our secretary's workload).

The Institute has a multimedia lab that it uses for creating distance learning courses, and I will do my proofs of concept in that lab.

The Institute's library is inadequate. We will buy several books and subscriptions and place them in the library.

The Software Engineering server is overtaxed and under-administered. It is likely that we'll set up our website on a commercial ISP (such as Earthlink) because this looks at the moment to be less expensive and more reliable than using the in-house system. We will reappraise this on an ongoing basis.